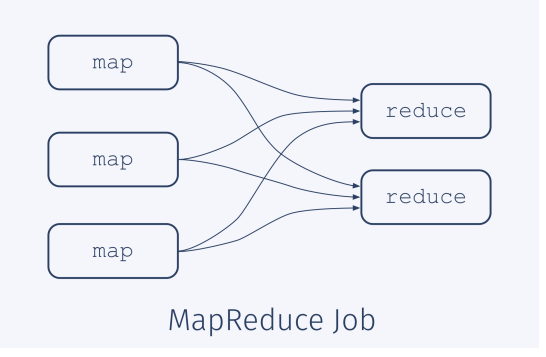
Characterizing and Synthesizing Task Dependencies of Data-Parallel Jobs in Alibaba Cloud

Huangshi Tian, Yunchuan Zheng, Wei Wang @ HKUST

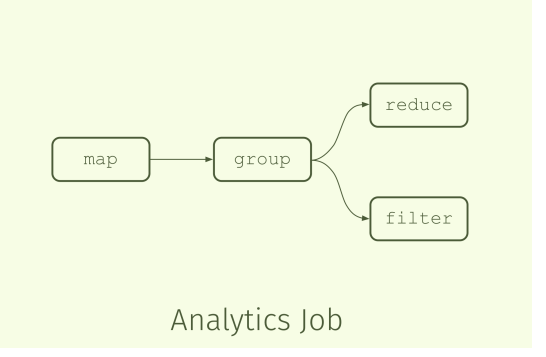
Nov. 21, 2019

Every job is born equal, but some are more complicated …

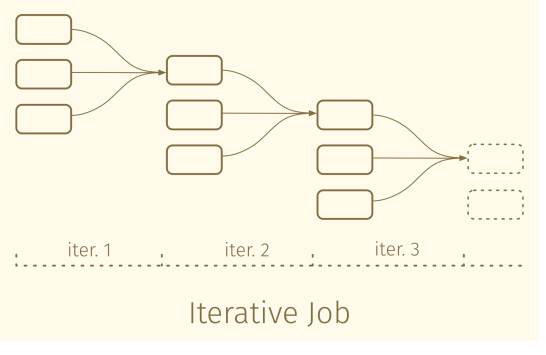
- **Hadoop**

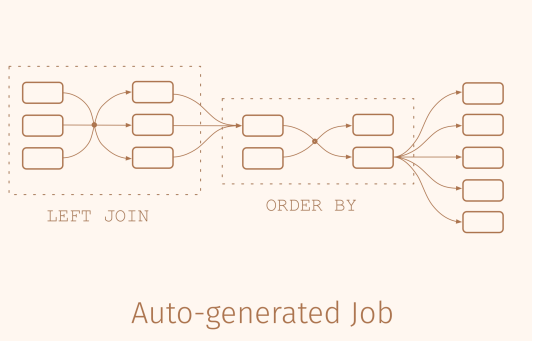


- **Spark**

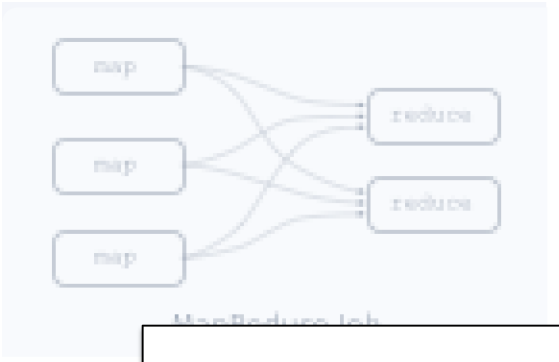


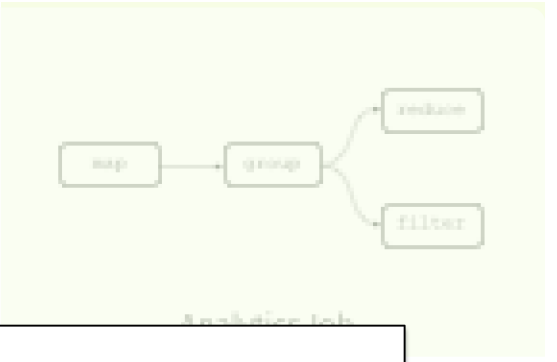
- **Ecosystem (MLlib, SQL, GraphX)**





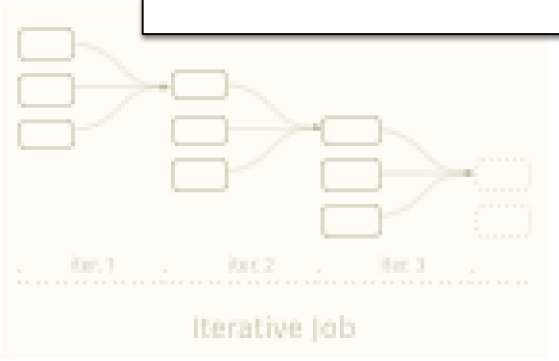
Do job DAGs have anything special?

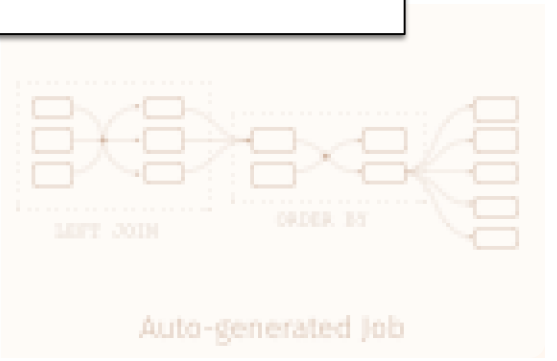




An (unsatisfactory) answer from the literature:

Production DAGs are *large and complex* ...

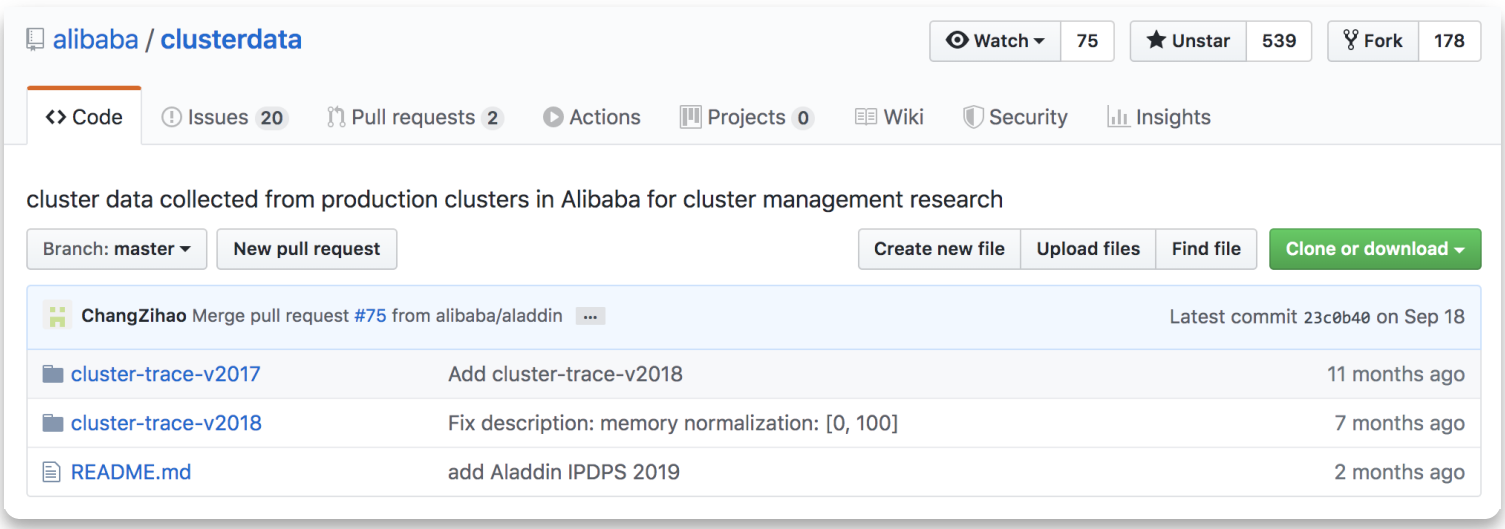




A Glimpse into Production Clusters

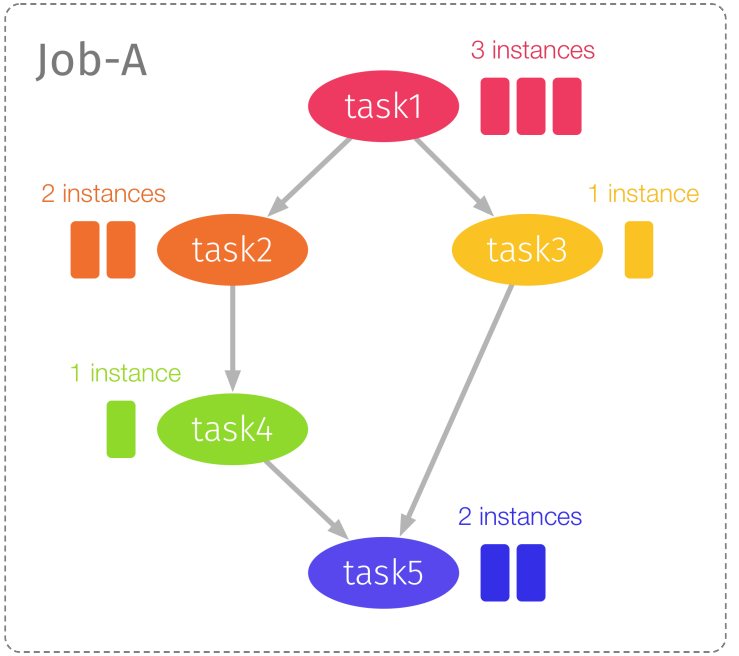
In the year of 2018, Alibaba has released a trace that ...

- spans **8 days**,
- records the activity of both **long-running containers** and **batch jobs** ...
- from **a cluster of 4034 machines**.



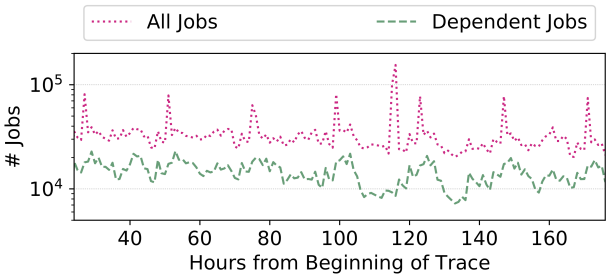
Zoom in on Batch Jobs

- Terminologies
  - task
  - instance
  - dependency
- Dataset Scale
  - 4.2M jobs
  - 14.3M tasks
  - 1.4B instances
- Applications
  - SQL queries (90%)
  - data analytics (10%)

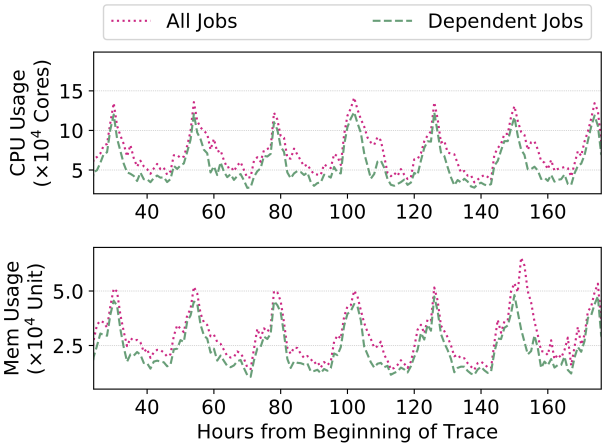


Overview of DAG Jobs

- Temporal Distribution

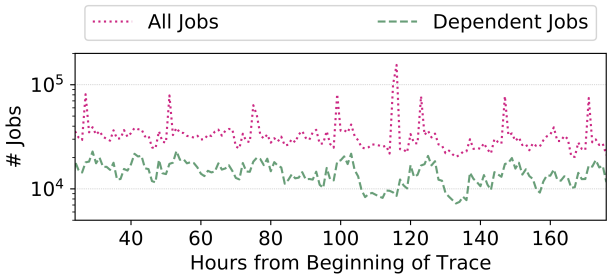


- Resource Consumption

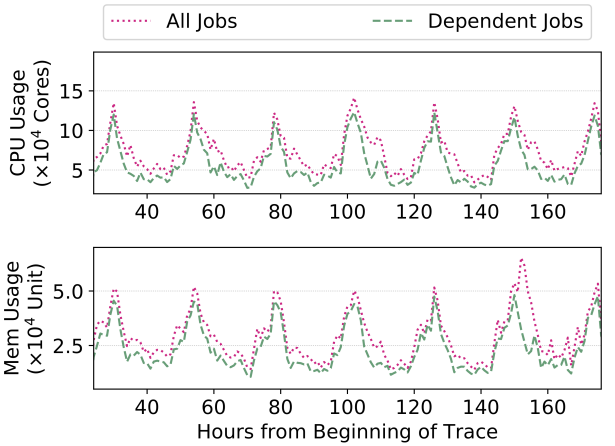


Overview of DAG Jobs

- Temporal Distribution



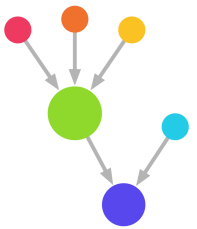
- Resource Consumption



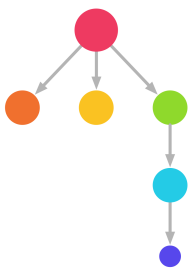
Takeaway: DAG jobs are prevalent and sometimes consume disproportionately many resources.

First Impression on Job DAGs: Trees Everywhere

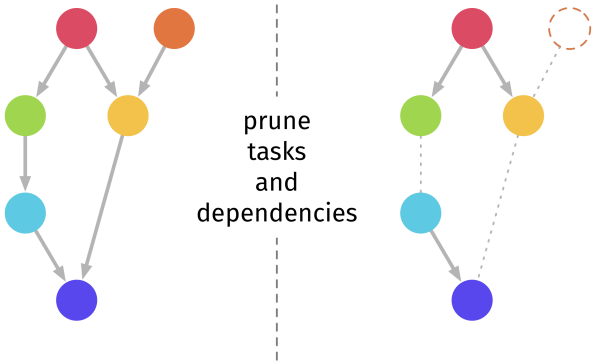
- 78.54% of all jobs are **gatter jobs**;



- 36.03% are **scatter jobs**;

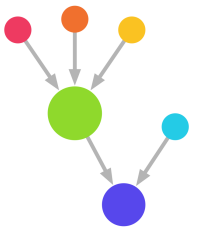


- Within **complex jobs**, 81.68% of tasks can be decomposed into scatter or gather jobs.

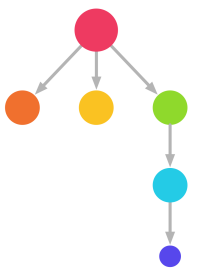


First Impression on Job DAGs: Trees Everywhere

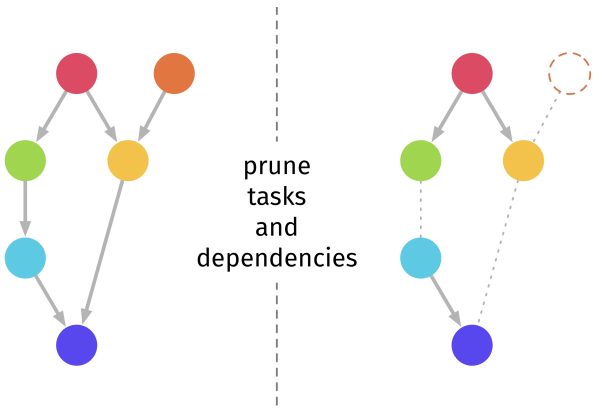
- 78.54% of all jobs are **gatter jobs**;



- 36.03% are **scatter jobs**;



- Within **complex jobs**, 81.68% of tasks can be decomposed into scatter or gather jobs.



Takeaway: There are opportunities for algorithmic scheduling.

Commonality or Peculiarity?

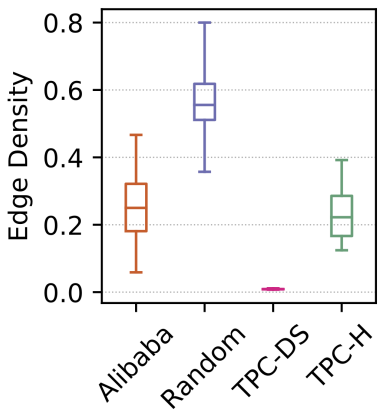
We introduce four datasets of DAGs for comparison:

1. **Alibaba DAGs** extracted from the trace,
2. **Random DAGs** generated by a uniformly random algorithm,
3. **TPC-DS DAGs** from the namesake benchmark,
4. **TPC-H DAGs** similar as above.

Sparsity and Probable Cause

- **Edge density** defined as:

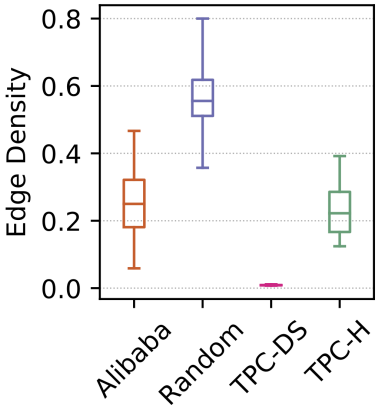
$$\frac{\text{\# dependencies}}{\text{\# possible dependencies}}$$



Sparsity and Probable Cause

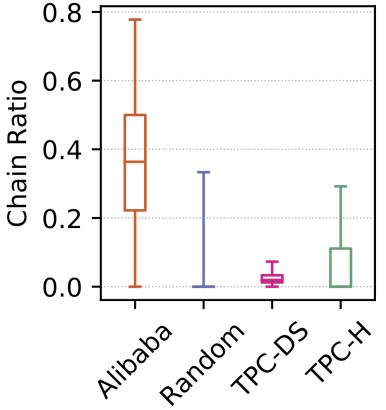
- **Edge density** defined as:

$$\frac{\text{\# dependencies}}{\text{\# possible dependencies}}$$



- **Chain ratio** defined as:

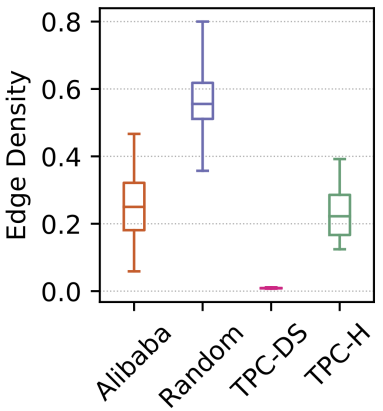
$$\frac{\text{\# tasks with only one parent/child}}{\text{\# all tasks}}$$



Sparsity and Probable Cause

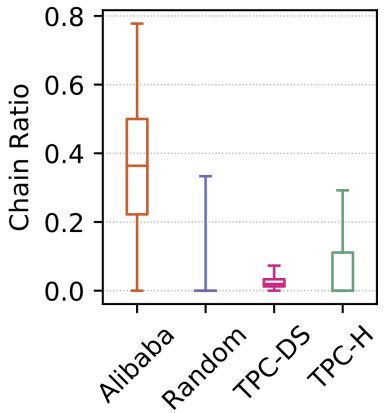
- **Edge density** defined as:

$$\frac{\text{\# dependencies}}{\text{\# possible dependencies}}$$



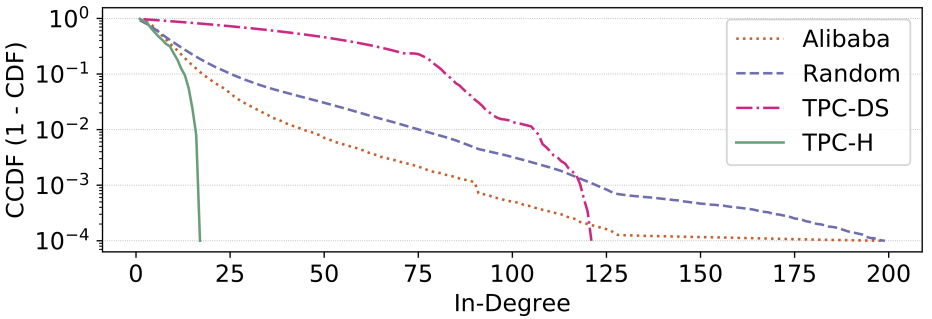
- **Chain ratio** defined as:

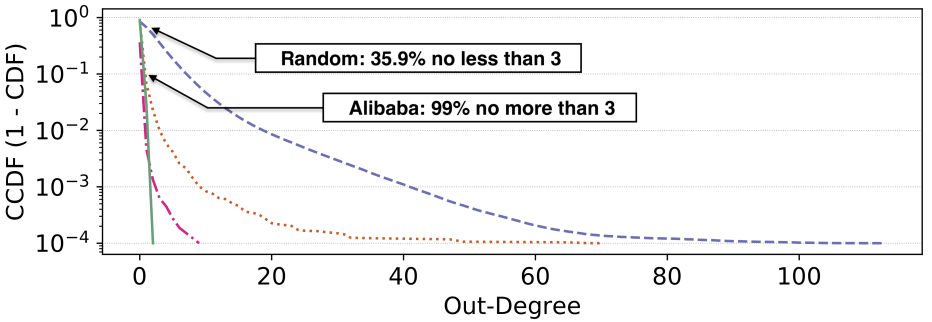
$$\frac{\text{\# tasks with only one parent/child}}{\text{\# all tasks}}$$



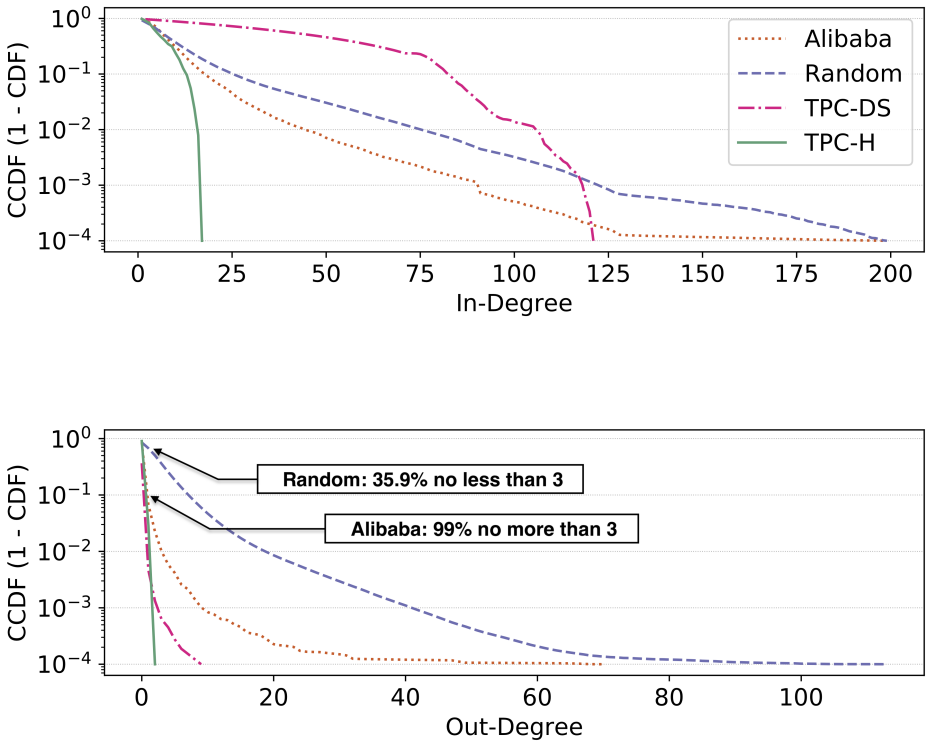
Takeaway: Job DAGs are sparse and have many chains.

In- and Out-Degrees





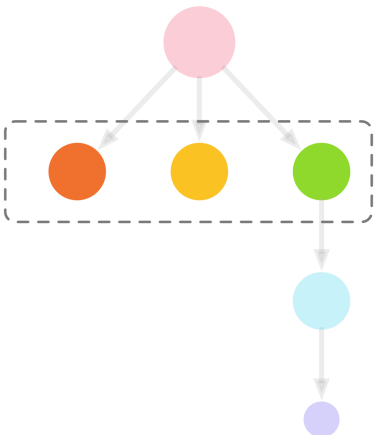
In- and Out-Degrees



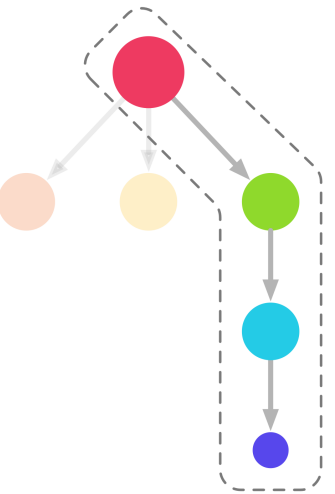
Takeaway: A task can have many dependencies, but typically a few children.

Shape of DAG

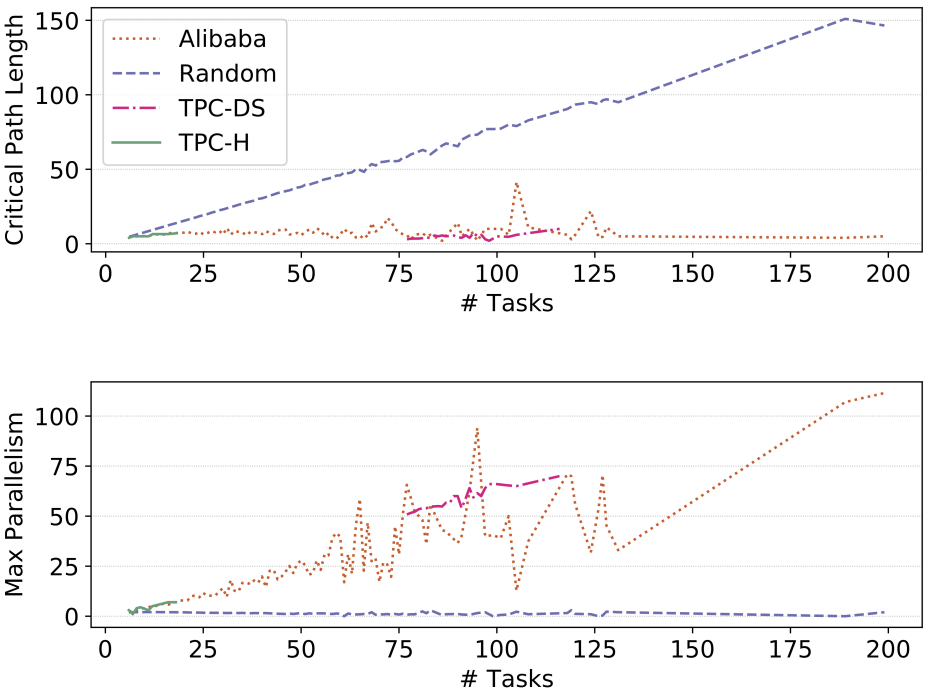
- Maximum Parallelism



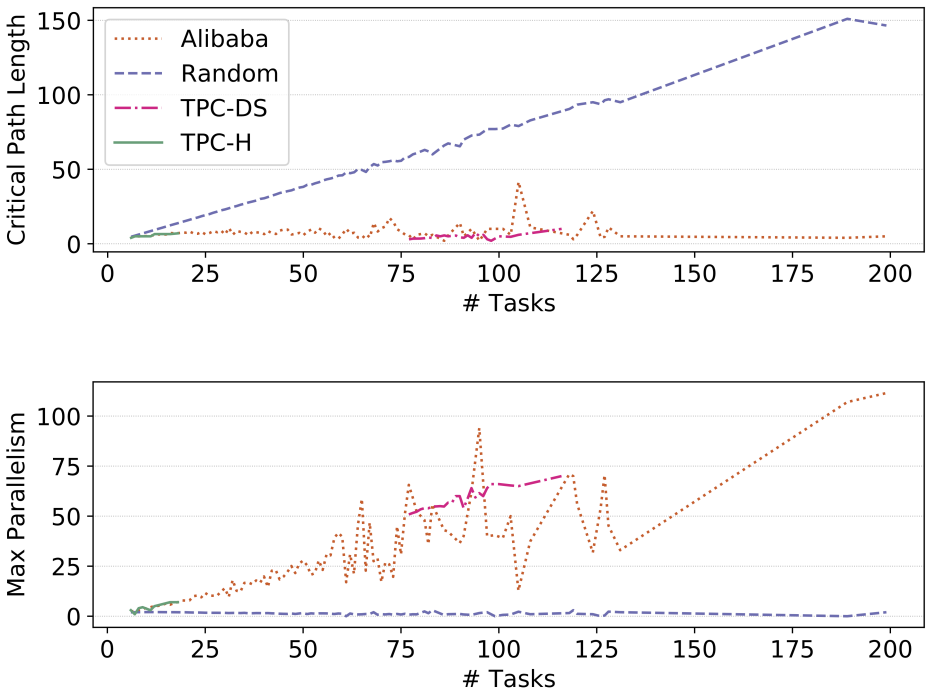
- Critical Path



Shape of DAG


Critical Path Length
150
100
50
0
Alibaba
Random
TPC-DS
TPC-H
0
25
50
75
100
125
150
175
200
Tasks
Max Parallelism
100
75
50
25
0
0
25
50
75
100
125
150
175
200
Tasks

Shape of DAG

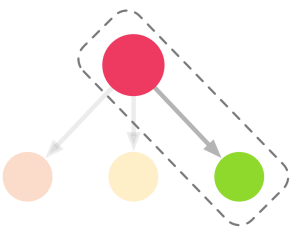


Takeaway: Production DAGs grow "wider" instead of "longer".

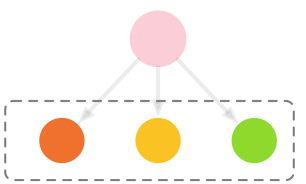
Runtime Performance of DAG Jobs

- **Runtime Variability**: troublemaker for cluster schedulers
  - straggler tasks
  - resource fragmentation

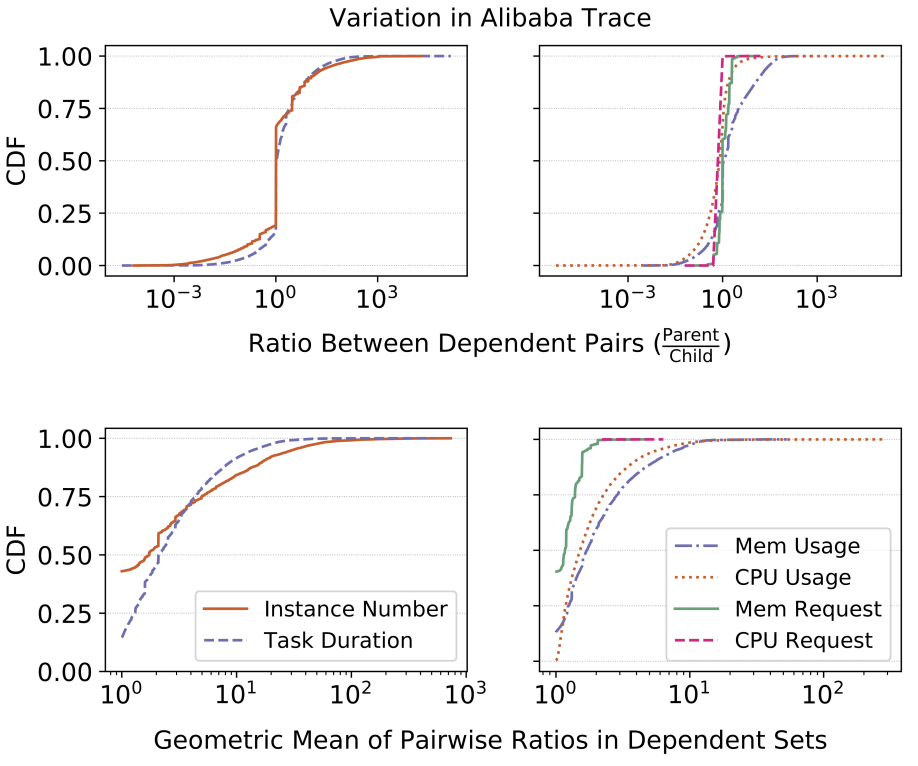
- **Measuring Variation**
  - dependent pair: ratio between metrics



  - dependent set: geometric mean of all pairwise ratios

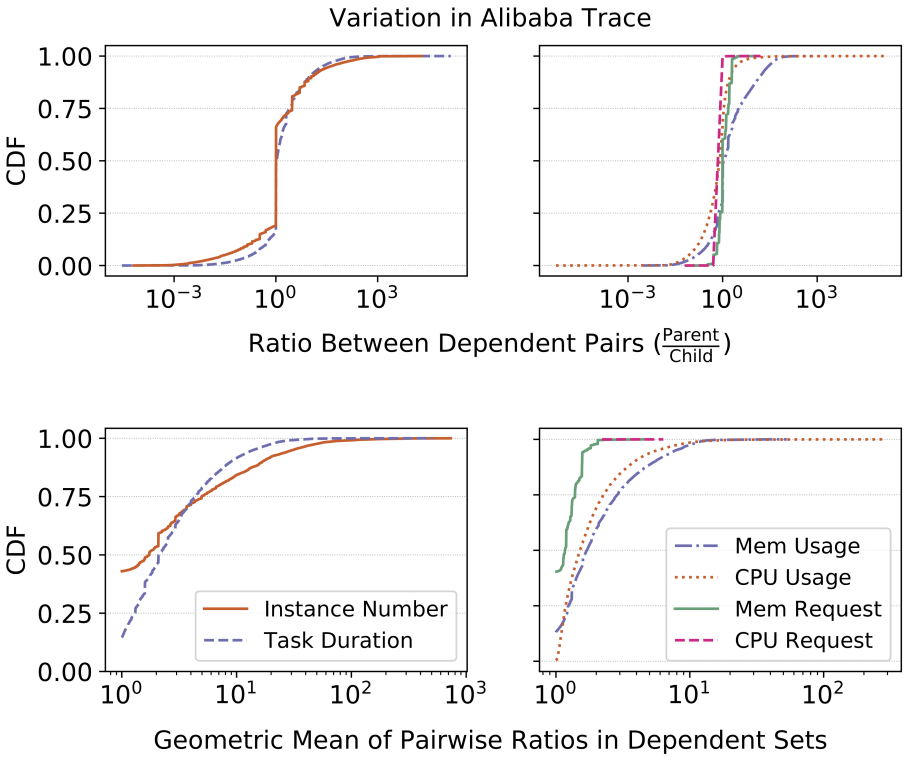


Does Dependency Constrain Runtime Variability?



... vary over 5x	Proportion
Instance #	26.46%
Duration	20.77%
CPU Usage	1.89%
Memory Usage	20.12%

Does Dependency Constrain Runtime Variability?

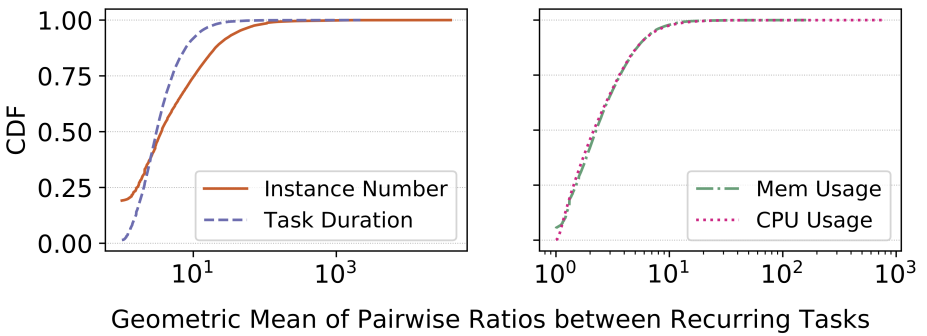


... vary over 5x	Proportion
Instance #	26.46%
Duration	20.77%
CPU Usage	1.89%
Memory Usage	20.12%

Takeaway: Unfortunately, not that much.

Variability of "Recurrent" Jobs

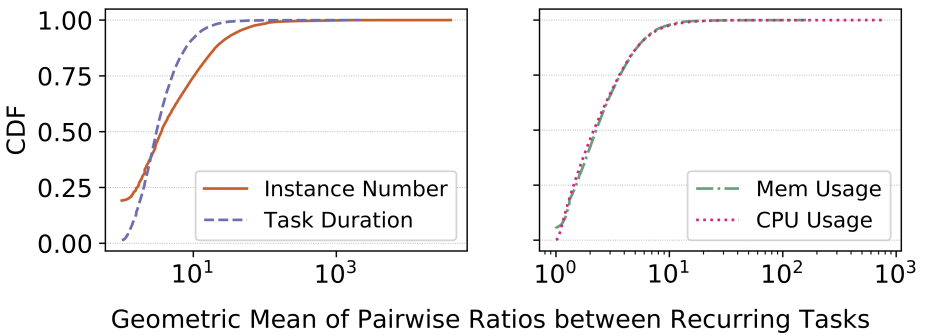
We select "recurrent" jobs by the criteria of (1) **isomorphic structures**, (2) **periodic submission intervals** and (3) **identical resource requests**.



... vary over 2x	Proportion
Instance #	69.25%
Duration	75.69%
CPU Usage	54.15%
Memory Usage	57.61%

Variability of "Recurrent" Jobs

We select "recurrent" jobs by the criteria of (1) **isomorphic structures**, (2) **periodic submission intervals** and (3) **identical resource requests**.



... vary over 2x	Proportion
Instance #	69.25%
Duration	75.69%
CPU Usage	54.15%
Memory Usage	57.61%

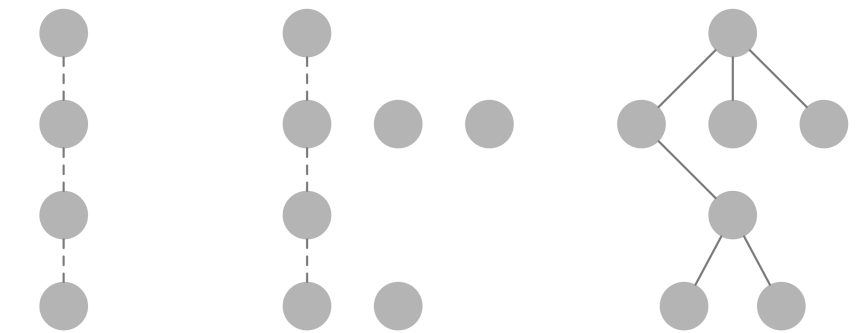
Takeaway: Recurrent tasks can have high variability.

How to Synthesize a DAG

STEP I: Randomly draw a critical path length from the distribution.

STEP II: Randomly decide how tasks are distributed along the path.

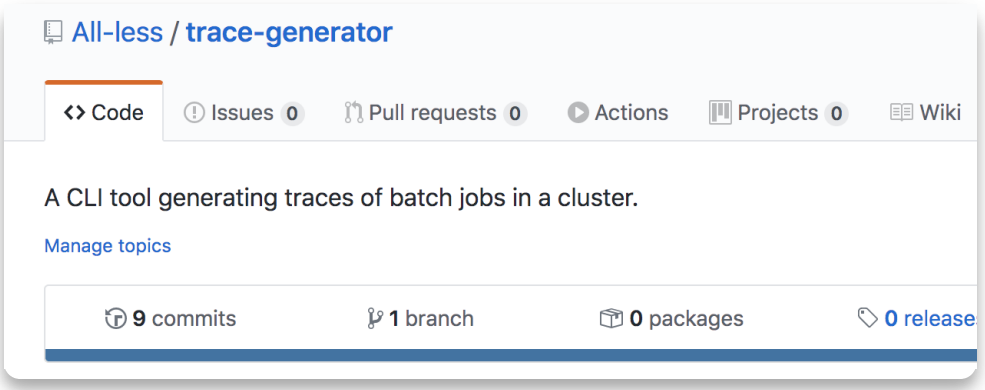
STEP III: Randomly connect tasks on adjacent levels.



(Please refer to the paper for the evaluation results.)

Trace Generator

- No need to manipulate 200GB+ of raw data.
- Flexibly control the duration, load and heterogeneity of the trace.



Summary

- Structural Properties of Job DAGs, ...
  - sparse
  - "bounded" critical path
  - increasing parallelism
- Runtime Performance, ...
  - salient variability
  - ... even among recurrent tasks
- and Trace Generator